下载APP

21 | 多传感器集成：浇花怎么实现自动化?

2020-12-28 郭朝斌

物联网开发实战 进入课程 >

讲述：郭朝斌
时长 10:22 大小 9.50M

你好，我是郭朝斌。

在前几讲的实战中，我们已经完整地实现了一个智能家居的照明场景。你不但可以用手机小程序控制智能电灯的开关、亮度和颜色，也能让智能电灯根据光照强度自动地打开和关闭，甚至你还可以语音来操控它。

在这个过程中，你应该已经掌握了基于硬件开发板快速地开发智能硬件的方法。这一讲中，我们就围绕**自动浇花**，做一个新的智能家居场景解决方案，同时也巩固一下你在前几讲学习到的硬件开发知识。

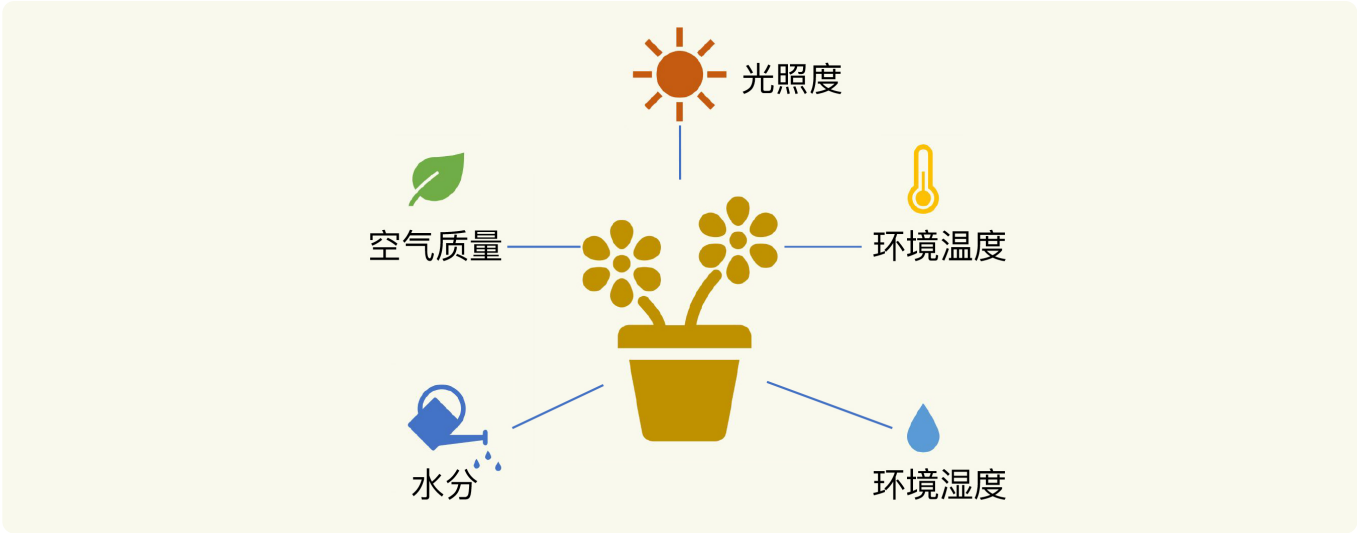
研究场景需求

在设计物联网产品的时候，我们需要先研究场景需求，明确监控指标，再根据这些指标安排合适的传感器。

比如自动浇花这个场景，很明显是为了自动控制水泵，及时给植物补充**水分**。毕竟水不仅是植物体的主要成分，支撑着细胞、维持着植物形态，而且也是传输营养物质的重要载体。所以，我们要监控的最重要的指标就是**土壤湿度**。

不过，用户的目的并不只是浇水而已，他们真正想要的，是看到自己养的植物健康地生长，所以我们最好把其他相关的数据也提供给他们。现在很多智能体重秤都会同时检测你的体脂率等其他反映身体健康的指标，也是一样的道理。

那么，植物生长还需要什么条件呢？



首先是**环境温湿度条件**，热带的植物无法忍受低温和干燥的环境条件，而温带植物遇到高温，也可能出现热衰竭的现象，所以环境的**温度**和**湿度**最好也能监控。

其次是**光照条件**，毕竟要植物进行光合作用嘛，所以**光照强度**需要监控。

最后，**空气质量**对植物的生长来说也很重要，比如**二氧化碳**是否充足，**有害气体**是否超标等。

不过考虑到家居环境，空气质量在日常种植过程中一般没有问题，不需要特别关注。当然，你也可以作为扩展任务来尝试一下。

所以综合来看，我们需要监控的指标包括土壤湿度、环境温湿度和光照强度。相比于市面上其他只能控制浇水的产品，我们的产品考虑的场景就更加全面了。这也是自己动手（DIY）的好处。

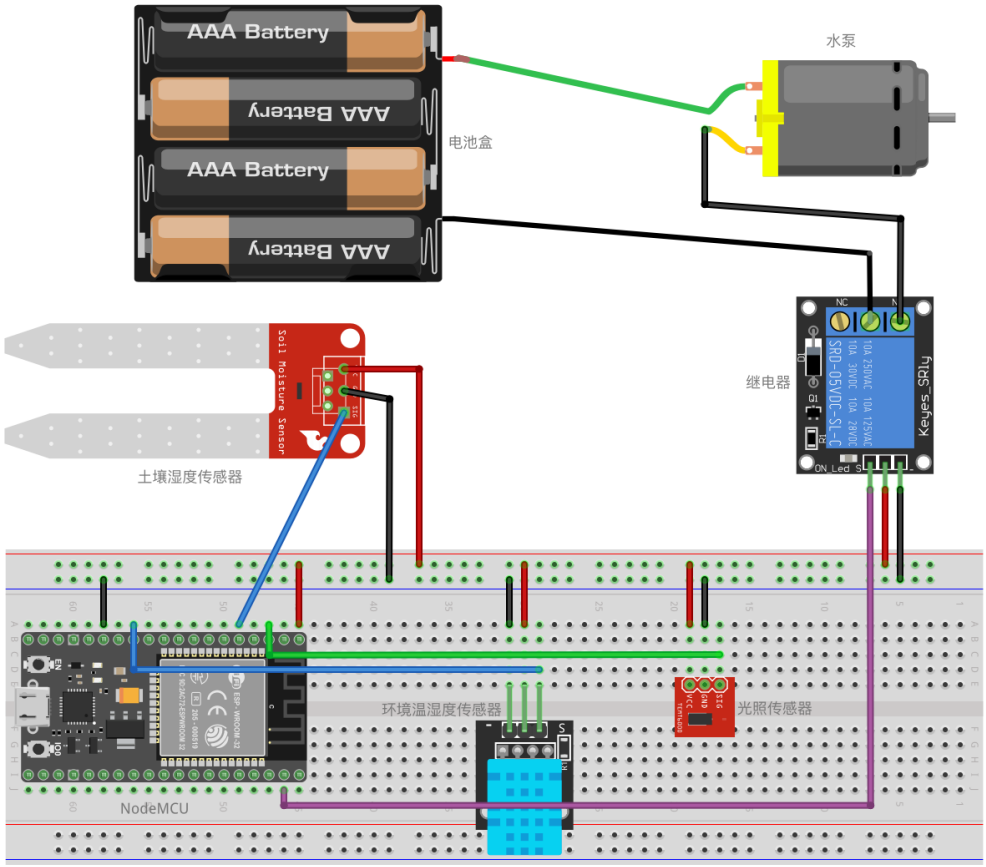
硬件电路

明确了监控指标之后，相应的传感器也就可以确定了，分别是土壤湿度传感器、环境温湿度传感器和光照传感器。

下面，我总结一下自动浇花器需要用到的材料：

1. NodeMCU ESP32 开发板。不过，这里我们可以使用 Wi-Fi 来连接网络，而不是第 18 讲中用到的蓝牙技术。因为在现实中，考虑到自动浇花器的工作环境，连接电源还是比较常见的。
2. 继电器，用于控制水泵的供电电路的通断。
3. 水泵，用于从水箱中抽水，并送到花盆中。
4. 电池盒，用于给水泵供电。
5. 土壤湿度传感器，它可以测量花盆土壤的湿度，然后输出模拟信号。
6. 环境温湿度传感器，它基于 DHT11 传感器，用于测量房屋中的温度和湿度，并且输出数字信号。它采用单总线（1-wire）接口与 NodeMCU 连接。
7. 光照传感器，它可以测量花盆位置接收到的光照条件。每种植物有喜阳光、耐阴等不同的特性，光照度和一段时间的光照累积量可以给我们提供参考，以便更好地满足植物的光照需求。
8. 面包板和杜邦线。

这里，我给出了一个电路图，方便你了解整个自动浇花器的设备组成。



软件实现

接下来，我们还是使用 Python 语言来完成软件的开发工作。

继电器

首先是继电器的控制。它和第 17 讲介绍的智能电灯的继电器控制方法没有本质区别，主要是改变了连接的 GPIO 管脚。根据硬件电路的连线，这里我们使用的是 GPIO23 管脚。

我把代码直接贴在文稿中，供你参考（它连接的管脚是 **GPIO23**）：

复制代码

```
## ！！！本文件采用商城的继电器模块FL-3FF-S-Z
## ！！！on(), off()状态相反。
## ！！！初始化中需要调用on()先关闭水泵电路
from machine import Pin

class Relay():
    ON = 0
    OFF = 1

    def __init__(self, pin):
```

```
        self.relaypin = Pin(pin, Pin.OUT)
        self.relaypin.on()
        self.last_status = self.OFF

    def set_state(self, state):
        tmp_state = self.ON if state==1 else self.OFF
        self.relaypin.value(tmp_state)
        self.last_status = tmp_state

    def state(self):
        return self.last_status
    def on(self):
        self.relaypin.value(self.ON)
        self.last_status = self.ON

    def off(self):
        self.relaypin.value(self.OFF)
        self.last_status = self.OFF
```

土壤湿度传感器

然后是土壤湿度传感器，我采用的是基于测量电阻值的变化来判断土壤水分含量的传感器模块。它会根据不同的电阻值，输出变化的模拟信息，数值越小，说明越干燥。使用的时候，你需要把它完全插入花盆的土壤中。

另外，还有一种基于土壤湿度变化，引起电容值变化的原理设计的土壤湿度传感器，你也可以考虑使用。它的好处是长期使用过程中不易腐蚀，因为没有裸露金属。

我把代码直接贴在文稿中，供你参考（它连接的管脚是 **GPIO34**）：

复制代码

```
from machine import ADC
from machine import Pin

class SoilSensor():

    def __init__(self, pin):
        self.sensor = ADC(Pin(pin))

    def value(self):
        value = self.sensor.read()
        print("Sensor ADC value:",value)
        return int(value*100/4095)
```

环境温湿度传感器

我刚才介绍过，环境温湿度传感器是基于 DHT11 实现的，并且使用的是单总线的连接方式。不过代码的开发过程并不复杂，因为 MicroPython 已经为 ESP32（也包括 ESP8266）实现了 DHT 代码组件。我们可以直接使用。

注意，DHT11 的单总线协议是奥松电子的自定义协议，它与 Dallas 半导体公司的 OneWire 协议是不同的。

代码我贴在下面，供你参考（它连接的管脚是 **GPIO14**）：

复制代码

```
import dht
from machine import Pin

class EnvSensor():

    def __init__(self, pin):
        self.sensor = dht.DHT11(Pin(pin))

    def value(self):
        self.sensor.measure()
        return (self.sensor.temperature(), self.sensor.humidity())
```

光照传感器

至于光照传感器，我们还可以继续使用第 18 讲的模组，所以我就不过多介绍了。

我直接贴出代码，供你参考（它连接的管脚是 **GPIO36**）：

复制代码

```
from machine import ADC
from machine import Pin

class IllumSensor():

    def __init__(self, pin):
        self.sensor = ADC(Pin(pin))

    def value(self):
        value = self.sensor.read()
        print("Sensor ADC value:",value)
```

```
12          return int(value/4095*600
```

完成联网开发

为了实现自动浇花器的联网控制，我们仍然需要将它接入腾讯云物联网平台。不过，我希望你能尝试独立完成这个工作。完成之后，你一定会感受到自己的成长。如果实现过程中有困难，可以回头参考第 17 讲的内容。

这里，我提供一下自动浇花器的物模型 JSON 文件，供你参考。

复制代码

```
{
    "version": "1.0",
    "profile": {
      "ProductId": "你的ProductID",
      "CategoryId": "909"
    },
    "properties": [
      {
        "id": "power_switch",
        "name": "水泵开关",
        "desc": "控制水泵启动关闭",
        "mode": "rw",
        "define": {
          "type": "bool",
          "mapping": {
            "0": "关",
            "1": "开"
          }
        }
      },
      {
        "id": "water_shortage",
        "name": "缺水状态",
        "desc": "水箱是否缺水",
        "mode": "r",
        "define": {
          "type": "bool",
          "mapping": {
            "0": "否",
            "1": "是"
          }
        }
      },
      {
        "id": "humidity",
```

```
        "name": "土壤湿度",
        "desc": "当前的土壤湿度",
        "mode": "r",
        "define": {
          "type": "int",
          "min": "0",
          "max": "100",
          "start": "0",
          "step": "1",
          "unit": "%"
        }
      },
      {
        "id": "env_temp",
        "name": "环境温度",
        "desc": "空间环境的温度",
        "mode": "r",
        "define": {
          "type": "float",
          "min": "-40",
          "max": "100",
          "start": "0",
          "step": "0.1",
          "unit": "℃"
        },
        "required": false
      },
      {
        "id": "env_hum",
        "name": "环境湿度",
        "desc": "周围环境的湿度",
        "mode": "r",
        "define": {
          "type": "int",
          "min": "0",
          "max": "100",
          "start": "0",
          "step": "1",
          "unit": "%"
        },
        "required": false
      },
      {
        "id": "env_illum",
        "name": "环境光照度",
        "desc": "周围环境的光照度",
        "mode": "r",
        "define": {
          "type": "int",
          "min": "0",
          "max": "6000",
          "start": "0",
```

```
          "step": "1",
          "unit": "lux"
        },
        "required": false
      }
    ],
    "events": [],
    "actions": []
  `
```

需要说明的一点是，其实 DHT11 的测温范围只有 0～50℃，但这里设定的环境温度范围是 -40℃～80℃。之所以这么设定，是因为另外一款温湿度传感器 DHT22 可以达到这个范围，这样当你想根据不同的环境条件灵活地调整温湿度传感器模块的时候，就不需要修改物模型了。

多传感器融合

在刚才的开发工作中，我们只使用了一个土壤湿度传感器来判断土壤的水分含量。先不考虑传感器本身的测量误差，单纯只测量土壤中一个位置的湿度数值，其实是不能准确反映整片土壤的水分情况的。

所以，我们可以使用多个土壤传感器进行测量，然后根据多个测量值来计算出更可靠的土壤水分含量。

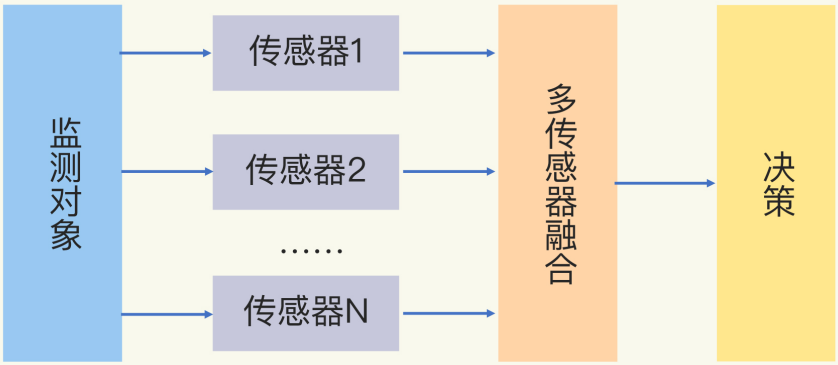
这个计算过程就是**多传感器融合**（Multi-Sensor Fusion，MSF）。

它不仅在自动驾驶这样的前沿领域中成为了保证决策正确的关键技术，而且在日常生活中也已经有了广泛的应用，比如手机上的地理位置定位。

我们知道，GPS 是通过卫星进行定位的，但是它的精度受到很多因素的影响。为了提高定位的精度，现在我们的手机上就普遍采用多传感器融合的方法，通过 GPS、蜂窝通信网基站和 Wi-Fi 热点的数据来综合计算，得到更准确的地理定位信息。

多传感器融合最关键的地方，不是多个传感器的**硬件连接**或**数据收集**，而是**融合算法**，也就是怎么进行多个不同维度的参数的处理，从而得到一个相对准确的、有意义的数据信息。

最基本的算法就是计算多个数值的加权平均值。适应性更广的算法有卡尔曼滤波和多贝叶斯估计等方法。



在条件允许的情况下，你可以尝试一下这个方法。毕竟 NodeMCU 丰富的 ADC 接口为我们提供了不错的实验条件。比如，你可以基于两个土壤湿度传感器，使用加权平均值算法优化自己的自动浇花器。

小结

总结一下，这一讲中，我介绍了自动浇花器的硬件电路参考设计和代码。你需要关注的主要内容有：

1. 自动浇花器主要是通过检测土壤湿度来判断水分含量，然后基于水分含量来控制水泵电源通断来实现自动浇花的目的。
2. 环境温湿度传感器采用的是 DHT11 模组，它采用单总线的方式与控制芯片通信，这是奥松电子的自定义协议。不过，基于 MicroPython 开发的时候，我们不需要自己开发驱动，可以直接使用 MicroPython 提供的 DHT 库。
3. 联网的开发，我们可以继续基于腾讯云物联网平台来实现。我提供了物模型，希望你可以基于智能电灯的开发经验独立完成这个任务。
4. 多传感器融合是提高决策正确性的重要方法，其中融合算法是最关键的。

在实战篇，我们主要基于开源的开发板硬件来完成智能硬件的开发工作。这在工作中有很大的用处，因为当我们有新的产品想法时，可以快速地进行技术验证，甚至用户反馈。

这也就是**最小可行性产品**（Minimum Viable Product，MVP）方法论。在软件领域，这种方法比较常见，而基于硬件开发板，我们可以很好地验证智能硬件产品的可行性，为企

业节约成本、降低风险。

思考题

最后，我给你留一个思考题吧。

在我提供的物模型里面，你应该可以看到“缺水状态”的属性。它可以用于指示水泵连接的水箱是不是已经缺水了，如果水箱水位不足，就会提醒我们及时给水箱补水。请你想一想，要怎么实现这个水位检测功能呢？请用文字描述一下你的思路。

欢迎你在留言区和我交流，也欢迎你将这一讲分享给你的朋友，大家一起交流学习。

提建议

© 版权归极客邦科技所有，未经许可不得传播售卖。 页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 20 | 智能语音：好玩的语音控制是怎么实现的？

下一篇 22 | 掌控数据：家里的数据可以怎么利用？

精选留言 (5)

写留言

978!6

2020-12-28

提一些可能性的方案 不考虑低功耗 设计寿命 安装难度等因素
连通器原理，在密壁的空间中安装一浮漂
1. 利用浮漂驱动一些微动开关 磁敏等
2. 顶端使用超声波等测距传感器 检测浮漂距离顶端的距离
3. 水箱顶端安装两个电极 浇花的水很难做到不导电 所以在水中导通，空气中短路...

展开

作者回复: 大赞👍

7

小人物
2020-12-31

希望老师能把配套硬件的官方技术文档地址给一下，比如温度传感器，配套有两种，引脚的作用不是很懂，小白不知道怎么去找资料看

展开

5

Nior
2020-12-30

直接利用现有的土壤湿度传感器，打开水泵后，土壤湿度没有发生比较明显的变化就是没水了

3

Allen
2020-12-28

请问示意电路图是什么软件，看着很漂亮

展开

作者回复: fritzing

1 2

评分家
2021-01-07

再加一个压力计？哈哈

展开

作者回复: 可以😁